AMENDMENTS TO THE SPECIFICATION

Please amend paragraph [0001] as follows:

[0001] This patent application claims the benefit of U.S. Provisional Patent Application Nos. 60/471,284, now expired, and 60/471,567, now expired, which were both filed on May 16, 2003 and entitled "DISTRIBUTED DOCUMENT OBJECT MODEL" and are both incorporated herein by reference in their entireties.

Please amend paragraph [0004] as follows:

[0004] The DOM is an application programming interface ("API") specification established by the World Wide Web Consortium ("W3C"). The W3C defines the DOM as "a platform- and language-neutral interface that will allow programs and scripts to dynamically access and update the content, structure and style of documents." The World Wide Web consortium's ("W3C") web site, w3c.org, provides a DOM specification. (~~<www.w3c.org/DOM>.~~) The DOM presents a programming interface for well-formed XML documents, including valid HTML, and defines how to manipulate a Document Object, such as an XML document. Using the DOM, a software program can, e.g., create a document, navigate its structure, and add, retrieve, modify, or delete its contents.

Please amend paragraph [0008] as follows:

[0008] The DOM defines objects, methods, properties, and events. As an example, the DOM defines a "Document" object that has a "getElementById" method. An example property of an object is "nodeValue" and an example event is "DOMNodeInserted." One skilled in the art will understand that an object model such as the DOM would have multiple objects, methods, properties, and events, and would further understand what they are used for and how they interrelate. The remainder of this specification assumes a baseline understanding of the current XML and DOM art beyond what is described above. This baseline is defined by the W3C DOM specifications, which include Document Object Model

(DOM) Level 2 Core Specification Version 1.0 (W3C Recommendation November 13, 2000), Document Object Model (DOM) Level 2 Events Specification Version 1.0 (W3C Recommendation November 13, 2000), Document Object Model (DOM) Level 2 HTML Specification Version 1.0 (W3C Recommendation January 9, 2003), Document Object Model (DOM) Level 2 Style Specification Version 1.0 (W3C Recommendation November 13, 2000), Document Object Model (DOM) Level 2 Views Specification Version 1.0 (W3C Recommendation, November 13, 2000), and Document Object Model (DOM) Level 2 Traversal and Range Specification Version 1.0 (W3C Recommendation November 13, 2000). These specifications are available at the W3C's web site ~~<http://www.w3.org/DOM>~~ (last visited October 1, 2003) and are all hereby incorporated herein by reference.

Please amend paragraph [0076] as follows:

[0076] Turning now to the figures, Figure 3 is a block diagram illustrating components of an embodiment of the DDOM system. The system 300 includes one or more clients 302A, 302B, a server 304, and a network 306. One skilled in the art will recognize that even though a single network and server are illustrated, there may be multiple networks, sub-networks, or servers in the system. As an example, a client may be behind a firewall in an intranet system, yet be communicating over an Internet to the server, which in turn may be on a separate intranet. The client and server may be any of a variety of forms of computing systems. As examples, a client may be a personal computer, personal digital assistant, advanced cellular telephone, or a pocket computing device. As further examples, the server may be a personal computer, mainframe computer, or minicomputer. One skilled in the art will recognize that computing devices of different forms and on separate communication networks are capable of communicating with one another to send or retrieve various forms of data. Each DDOM client component 308 of a client has a DDOM document that is a copy of the server's master version of the document. This master version is handled by the DDOM server component 310 of the server. The master version of the document and the clients' copies are represented as tree structures. The

DDOM client and DDOM server components expose the DOM API and DDOM's extensions to the API. In this embodiment, the DDOM client components, DDOM server component, and network comprise the DDOM system.

Please amend paragraph [0082] as follows:

[0082] A DDOM client may send a Mutation Request 418 to the DDOM server based on a mutation made by a user of the client software program. When the Mutation Request from the client arrives at the server, the DDOM Protocol Adapter and Message Layer of the server may transform the arriving packets or frames into a form acceptable by the DDOM server.

Please amend paragraph [00129] as follows:

[00129] At block 1066, the routine applies the received mutation on the client's local copy of the document. Either after blocks 1064 or 1066, the routine may clear information for the target of the mutation in the local history log and adjust the history log's records that refer to the target node as its positioning is now established based on the message from the server. This may be the same history log that may be used to adjust the target location at block 1064. At block 1068, the routine can generate a post-application event that indicates that a mutation was applied. The routine returns execution to its caller at block 1070.

Please amend paragraph [00136] as follows:

[00136] Figure 13 illustrates a flow diagram for an embodiment the Roll Back routine. The routine 1300 begins at step 1302 where it receives an indication of a requested version. At block 1304, the routine locates and loads a stored snapshot from Version Storage 414 that is near the desired version. Determination of proximity of version numbers is discussed above in relation to Figure 12. At block 1306, the routine undoes the effects of mutations between the loaded snapshot and the desired version. The routine ends at block 1308.

Please amend paragraph [00137] as follows:

[00137] Figure 14 illustrates a block diagram of an embodiment of a communications protocol stack used by the system. The stack 1400 includes a DDOM Message Layer 1402. The Message Layer defines message content and format and may communicate with a Universal Protocol Layer 1404. The Universal Protocol Layer offers an abstract interface to the message layer for transmission and reception of messages. The Universal Protocol Layer interacts with a number of possible protocol adapters that manage communications over transport protocols. The illustrated embodiment shows UDP 1407, TCP 1409, HTTP 1411, and HTTPS 1412. As an example, a UDP protocol adapter 1406 manages communications over UDP 1407. Similarly, the TCP protocol adapter 1408 manages communications over TCP/IP. Furthermore, the HTTP protocol adapter 1410 manages communications over HTTP and HTTPS. The communication system architecture illustrated here can be extended to support other transport protocols 1414 as required via protocol adapter 1413. The UDP and HTTP communication protocols use the IP and TCP/IP protocol layers, respectively. Other communication protocols may either use the TCP protocol layer directly or another protocol layer 1414. The stack presents a session-oriented reliable layer to either a client DDOM system or a server DDOM system. One skilled in the art will recognize that a protocol stack may be comprised of individual or multiple protocol forms.